



PATENT ABSTRACTS OF JAPAN

(11) Publication number: 10333905 A

(43) Date of publication of application: 18 . 12 . 98

(51) Int. Cl. **G06F 9/38**
G06F 9/38

(21) Application number: 09141687

(71) Applicant: KOFU NIPPON DENKI KK

(22) Date of filing: 30 . 05 . 97

(72) Inventor: TAKATO MASAHIKO

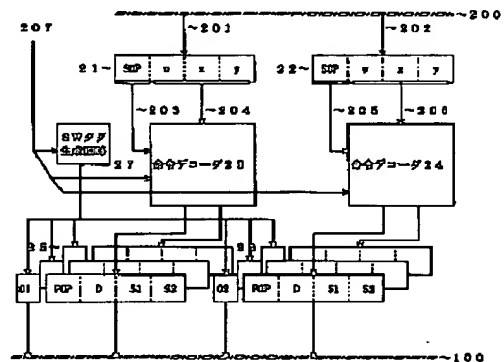
(54) **INFORMATION PROCESSOR FOR SUPER
 SCALAR SYSTEM**

COPYRIGHT: (C)1998,JPO

(57) Abstract:

PROBLEM TO BE SOLVED: To correctly control even the occurrence of even to be processed between software instructions by imparting a tag to a software instruction unit and recognizing the break of the software instruction.

SOLUTION: As for the software instructions stored in a software instruction queue, by an instruction issuance permission instruction 207, a first SW instruction is decoded by an instruction decoder 23, a second SW instruction is decoded by the instruction decoder 24 and they are developed to plural firmware instructions. At the time, the first SW instruction is developed to the two pieces of firmwares, the second SW instruction is developed to the three pieces of the firmwares and they are stored in firmware instruction queues 25 and 26. Further, an SW tag number 01 and the two firmware instructions are stored in the firmware instruction queue 25, the SW instruction tag number 02 and the three firmware instructions are stored in the firmware instruction queue 26 and the break of the software instruction is recognized.



(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平10-333905

(43)公開日 平成10年(1998)12月18日

(51)Int.Cl.⁹

G 0 6 F 9/38

識別記号

3 1 0

3 7 0

F I

G 0 6 F 9/38

3 1 0 F

3 7 0 A

審査請求 有 請求項の数 3 O L (全 6 頁)

(21)出願番号

特願平9-141687

(22)出願日

平成 9 年 (1997) 5 月 30 日

(71)出願人 000168285

甲府日本電気株式会社

山梨県甲府市大津町1088-3

(72)発明者 高遠 雅彦

山梨県甲府市大津町1088-3 甲府日本電
気株式会社内

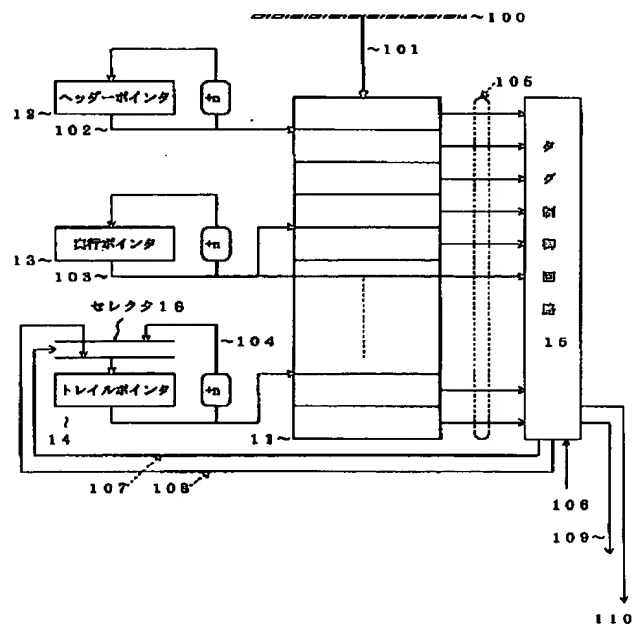
(74)代理人 弁理士 山下 稔平

(54)【発明の名称】 スーパースカラー方式の情報処理装置

(57)【要約】

【課題】 ソフトウェア命令間の区切りを認識でき、ソフトウェア命令間で処理しなければならないイベントが発生した場合でも正しく制御することを可能としたスーパースカラー方式の情報処理装置を提供する。

【解決手段】 ソフトウェア命令を複数のファームウェア命令に展開してプロセッサの1クロックサイクル中に前記複数のファームウェア命令を実行することが可能な情報処理装置において、前記ソフトウェア命令単位にタグ105を付与することで前記ソフトウェア命令の区切りを認識する手段15を具備したことを特徴とするスーパースカラー方式の情報処理装置。



【特許請求の範囲】

【請求項1】 ソフトウェア命令を複数のファームウェア命令に展開してプロセッサの1クロックサイクル中に前記複数のファームウェア命令を実行することが可能な情報処理装置において、

前記ソフトウェア命令単位にタグを付与することで前記ソフトウェア命令の区切りを認識する手段を具備したことを特徴とするスーパースカラー方式の情報処理装置。

【請求項2】 ソフトウェア命令を複数のファームウェア命令に展開してプロセッサの1クロックサイクル中に複数のファームウェア命令を実行することが可能な演算処理装置において、

前記ソフトウェア命令を複数のファームウェア命令に展開する際に、前記ソフトウェア命令毎に識別可能なSWタグを付与する回路27と、

前記ファームウェア命令毎に必要な登録内容とSWタグをinorderに登録するリオーダーバッファ11と、前記リオーダーバッファ11に登録したエントリの上限を示すヘッダーポインタ12と、

現在実行中の命令が格納されているエントリを示す実行ポインタ13と、

前記エントリの下限を示すトレイルポインタ14と、前記ソフトウェア命令間のイベント処理が発生した場合に、前記ソフトウェア命令の区切りを判別し、前記トレイルポインタを前記ソフトウェア命令の区切りに移動するタグ制御回路15と、を有することを特徴とするスーパースカラー方式の情報処理装置。

【請求項3】 前記タグ制御回路15からタグ情報110とキャンセル指示109を受け取り、各ステージ毎に持ち回っているSWタグ番号31、32と比較するタグチェック回路33、34を有することを特徴とする請求項2記載のスーパースカラー方式の情報処理装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、複雑なソフトウェア命令を単純なファームウェア命令に展開し、プログラム中に書かれている命令列とは異なる順番で実行する情報処理装置に関し、特に、スーパースカラー方式の情報処理装置に関する。

【0002】

【従来の技術】 従来、演算処理装置においては性能を上げるために複数の命令をオーバーラップして実行するパイプライン処理が行われている。

【0003】 データ依存のために、ある命令が実行できない場合でも、その命令よりも後の命令で実行できる命令があれば、その命令を先に実行することによってデータ依存による性能低下を避ける方法がある。この方法はプログラム中の命令列とは異なる順番で命令を実行する方法でout-of-order実行と呼ばれる。

【0004】 out-of-orderで実行した命令

の結果をそのままレジスタファイルに格納すると、例外が発生した場合に問題となる。例外が発生するとその例外のために完了できなかった命令のアドレスが退避され、例外の処理を行った後、退避された命令のアドレスから再び実行が開始される。

【0005】 ところが、out-of-order実行であるために、退避された命令アドレスの後続の命令が例外発生時にすでに実行を完了していたとすると、その命令は例外処理のあとに再び実行されてしまう。この問題を解決するためにリオーダーバッファが提案されている。

【0006】 特開平6-19707号公報の(13)頁に記載されている従来のリオーダーバッファのブロック図を用いて説明する。図7は、この従来のリオーダーバッファのブロック図であり、図7において、121はリオーダーバッファであり、データを格納するデータフィールド56とその格納先のレジスタ番号(デスティネーションレジスタ番号)を格納するデスティネーションレジスタフィールド55からなるエントリー122を複数持ち、out-of-orderで得られた演算器52の結果データを一旦格納し、これをプログラムに書かれている命令列の順番でレジスタファイル51にデータを格納するものである。

【0007】 リオーダーバッファ121に保持されたデータをレジスタファイル51に格納する前に、後続の命令がそのデータを必要とする場合、レジスタファイル51からの読み出しを禁止し、リオーダーバッファ121からデータを供給する必要がある。これは、リオーダーバッファ121の各エントリー122に依存検出回路57を設け、エントリー122のデスティネーションレジスタフィールド55に格納されているデスティネーションレジスタ番号と後続命令のソースレジスタ番号60及び61を比較することによってデータ依存が検出され、データ依存情報64、65によってエントリー122のデータフィールド56の出力が制御される。

【0008】 ここで、リオーダーバッファ121に同じデスティネーションレジスタ番号を持つエントリーが複数ある場合、最新データを選択しなければならない。そのため、通常、比較した結果、得られる複数の一致信号の中から優先度をつけて選ぶ回路を有している。

【0009】

【発明が解決しようとする課題】 第1の問題点は、従来の技術において、複雑なソフトウェア命令を単純なファームウェア命令に展開し、プログラム中に書かれている命令列とは異なる順番で実行する演算処理装置において、ソフトウェア命令実行中にイベント事象が発生した場合、ソフトウェア命令間でイベント処理を行うことができないことである。

【0010】 その理由は、ソフトウェア命令を複数のファームウェア命令に展開し、命令の実行順序もout-

10

20

30

40

50

of-orderであるためソフトウェア命令単位の区切りが判別できないからである。

【0011】〔発明の目的〕本発明の目的は、ソフトウェア命令単位のタグビットをリオーダーバッファに登録することにより、ソフトウェア命令間の区切りを認識でき、ソフトウェア命令間で処理しなければならないイベントが発生した場合でも正しく制御することを可能とした回路を提供することにある。

【0012】

【課題を解決するための手段】本発明は、前述した課題を解決するための手段として、ソフトウェア命令を複数のファームウェア命令に展開してプロセッサの1クロックサイクル中に前記複数個のファームウェア命令を実行することが可能な情報処理装置において、前記ソフトウェア命令単位にタグを付与することで前記ソフトウェア命令の区切りを認識する手段を具備したことを特徴とするスーパー scaler方式の情報処理装置を提供するものである。

【0013】また、ソフトウェア命令を複数のファームウェア命令に展開してプロセッサの1クロックサイクル中に複数個のファームウェア命令を実行することが可能な演算処理装置において、前記ソフトウェア命令を複数のファームウェア命令に展開する際に、前記ソフトウェア命令毎に識別可能なSWタグを付与する回路27と、前記ファームウェア命令毎に必要な登録内容とSWタグをin-orderに登録するリオーダーバッファ11と、前記リオーダーバッファ11に登録したエントリの上限を示すヘッダーポインタ12と、現在実行中の命令が格納されているエントリを示す実行ポインタ13と、前記エントリの下限を示すトレイルポインタ14と、前記ソフトウェア命令間のイベント処理が発生した場合に、前記ソフトウェア命令の区切りを判別し、前記トレイルポインタを前記ソフトウェア命令の区切りに移動するタグ制御回路15と、を有することを特徴とするスーパー scaler方式の情報処理装置を、上記手段とするものである。

【0014】また、前記タグ制御回路15からタグ情報110とキャンセル指示109を受け取り、各ステージ毎に持ち回っているSWタグ番号31、32と比較するタグチェック回路33、34を有することを特徴とするスーパー scaler方式の情報処理装置でもある。本発明のスーパー scaler方式は、イベント事象が発生した場合にソフトウェア命令間を認識し、ソフトウェア命令間で正しくイベント処理が行える回路を有することが特徴である。

【0015】具体的には、ソフトウェア命令毎に識別可能なSWタグを付与する回路27と、ファームウェア命令毎に必要な登録内容とSWタグ番号を登録するリオーダーバッファ11と、リオーダーバッファ11のエントリの上限を示すヘッダーポインタ12と、現在実行中の命

令が格納されているエントリを示す実行ポインタ13と、エントリの下限を示すトレイルポインタ14と、ソフトウェア命令の区切りを判別するタグ制御回路15を具備することを特徴とする。

【0016】

【作用】本発明によれば、ソフトウェア命令単位にSWタグ番号を付与し、ファームウェア命令の情報とともにSWタグ番号もリオーダーバッファに登録することで、現在実行中のソフトウェア命令を判断することができる。

10 【0017】これにより、イベント事象が発生した場合でもリオーダーバッファからSWタグ番号を抽出することで、ソフトウェア命令間を正しく判断することができるため、イベント処理を正常に行うことができる。

【0018】また、SWタグ番号を各ステージの情報と合わせて持ち回っており、タグ制御回路からキャンセル対象のSWタグを各ステージに通知し、各ステージでキャンセル対象のステージか否かを判断することで無効なデータのみをキャンセルすることができる。

【0019】

20 【実施例】次に、本発明の実施例について図面を参照して説明する。

【0020】〔構成〕図1は、本発明のリオーダーバッファの外部構成を示したブロック図である。

30 【0021】図1を参照すると、本発明の実施例は、ファームウェア情報、SWタグを登録するリオーダーバッファ11と、リオーダーバッファに登録されている最古のファームウェア命令を指すヘッダーポインタ12と、現在実行中のファームウェア命令を指す実行ポインタ13と、リオーダーバッファに次に登録する番地を指すトレイルポインタ14と、リオーダーバッファ11に登録されているSWタグ情報105とイベント通知106からトレイルポインタの制御情報を生成するタグ制御回路15から構成される。

【0022】図2は、ソフトウェア命令をファームウェア命令に展開するブロック図である。

40 【0023】ソフトウェア命令を格納するソフトウェア命令キュー21、22とソフトウェア命令のOPコードと各フィールド情報からファームウェア命令を生成する命令デコーダ23、24と、命令デコーダ23、24から生成されたファームウェア命令を格納するファームウェア命令キュー25、26と、実行部からの発行許可指示によりSW命令単位にタグを生成するタグ生成回路27から構成される。

【0024】図3は、各パイプラインキャンセル回路を示すブロック図である。

50 【0025】各ステージ毎にSWタグ番号31、32を格納する手段と、タグ比較を行うタグチェック回路33、34と、比較結果でキャンセル対象であることを認識した場合に、後述するバリッド35、36をクリアする論理積37、38から構成される。

【0026】〔動作〕次に、本発明の実施例の動作について、図2を参照して詳細に説明する。

【0027】ソフトウェア命令をファームウェア命令に展開するイメージを図4に示す。図4に示すように、1つのソフトウェア命令は、1～n個のファームウェア命令に展開される。図4の例だと、SW命令Aは2つのファームウェア命令に展開され、SW命令Bは3つのファームウェア命令に展開される。

【0028】また、SW命令タグ番号は、ソフトウェア命令単位に付与されるため、SW命令Aから展開されたファームウェア命令は同一のSW命令タグ番号“01”、SW命令Bから展開されたファームウェア命令は同一のSW命令タグ番号“02”が付与される。

【0029】ソフトウェア命令バス200には、命令フェッチ部から供給された複数のソフトウェア命令が存在する。ソフトウェア命令キュー21、22には、次に実行するソフトウェア命令をソフトウェア命令バス200より順次格納する。図2の場合、キューの数は2個としているが、いくつでもかまわない。

【0030】図4を例にすると、SW命令Aはソフトウェア命令キュー21に、SW命令Bはソフトウェア命令キュー22に格納される。

【0031】ソフトウェア命令キューに格納されているソフトウェア命令は、命令発行許可指示207によって、SW命令Aは、命令デコード23に、SW命令Bは命令デコード24によりデコードされ、複数のファームウェア命令に展開される。この場合、SW命令Aは2個の、SW命令Bは3個のファームウェアに展開され、ファームウェア命令キュー25、26に格納される。

【0032】また、命令発行許可指示207によりSWタグ生成回路27からソフトウェア命令単位にタグ番号が生成され、ファームウェア命令と合わせて、ファームウェア命令キュー25、26に格納される。SWタグ番号はソフトウェア命令毎に昇順に割り振られる。つまり、古い命令のSW命令タグ番号程、値が小さくなる。

【0033】以上の動作により、ファームウェア命令キュー25にはSWタグ番号“01”と2個のファームウェア命令が、ファームウェア命令キュー26にはSW命令タグ番号“02”と3個のファームウェア命令が格納されている。

【0034】ファームウェアキュー25、26に格納されたファームウェア命令とSWタグ番号はファームウェア命令バス100に順次送出される。

【0035】次に、図1を参照して詳細に説明する。ファームウェア命令バス100に送出されたファームウェア命令の情報とSWタグ番号はリオーダバッファ登録バス101を経由してリオーダバッファ11に登録される。

【0036】リオーダバッファ11に登録される内容の例を図5に示す。図5に示すように、リオーダバッファ

11には登録内容が有効であるか無効であるかを表すバリッドVと、ファームウェア命令で指定されているデスティネーションレジスタ番号と、ペンディング状態であるかを判定するビジーフラグBと、SW命令タグ番号と、ライトデータ、ステータス情報などが含まれている。これらの情報がファームウェア命令単位に各エンタリーに登録される。

【0037】リオーダバッファ11は、循環型FIFO構造になっており、各ポイントによって制御される。ヘッダーポイント12は、リオーダバッファ11に登録されている最古のエンタリー番号を示し、トレイルポイント14は、ファームウェア命令バス100から次に登録するエンタリー番号を示し、実行ポイント13は現在実行中であるファームウェア命令のエンタリー番号を示している。

【0038】図6に、図4のファームウェアがリオーダバッファ11に登録されているイメージ図を示す。各ファームウェア命令の情報はエンタリー番号20～24に登録されており、SW命令Aの1つめファームウェア命令が既に実行が完了しているとする。この場合、エンタリー番号20のステータスには終了(Done)が表示される。その他の命令はまだ実行が完了していないためビジーフラグBは“1”であり、ステータスは実行中(exec)、実行待ち(Ready)、ペンディング状態(Wait)となっている。この状態では、トレイルポイント14は“25”、実行ポイント13は“21”を指している。

【0039】この時にイベント通知106が通知されると、タグ制御回路15は現在実行中である命令のエンタリー番号を実行ポイント13から実行ポイント番号103を経由して受け取る。実行ポイント13は“21”を指しているため、リオーダバッファ11のエンタリー21に登録してあるSWタグ番号“01”をSWタグ情報105から受け取る。タグ制御回路15でエンタリー21のSWタグ番号と同一の番号をもつエンタリーを検索し、その中で最も新しく登録されたエンタリー番号をさらに検索する。

【0040】検索した結果のエンタリー番号（この場合は“21”）に+1インクリメントした値をトレイルポイント番号108にセットし、トレイルポイントセット指示107を“1”にする。この指示によりセレクト16は、トレイルポイント番号108を選択し、トレイルポイント14を、現在指しているエンタリー番号“25”から“22”に変更する。

【0041】また、タグ制御回路15はキャンセル指示109を発行するとともにキャンセル対象のタグ番号である現在実行中のSWタグ番号“01”に+1インクリメントした値“02”をタグ情報110に送出する。

【0042】次に図3を用いてSWタグのキャンセル動作を説明する。

【0043】各ステージには、ファームウェア命令を実行するための情報以外に、SWタグ番号31、32と一緒に持ち回っており、キャンセル指示109とタグ情報110を受け取ると自ステージのSWタグ番号31、32と比較を行い、自ステージのSWタグ番号がタグ情報110で通知された番号よりも大きければ自ステージの情報は無効であるため、対応するステージのバリッドを論理積37、38により“0”に落とす。

【0044】

【発明の効果】第1の効果は、複雑なソフトウェア命令を単純なファームウェア命令に展開し、プログラム中にかかっている命令列とは異なる順番で実行する演算処理装置において、イベント処理が正しく行える。

【0045】その理由は、ソフトウェア命令単位にタグを付与することでソフトウェア命令の区切りを認識することができるからである。

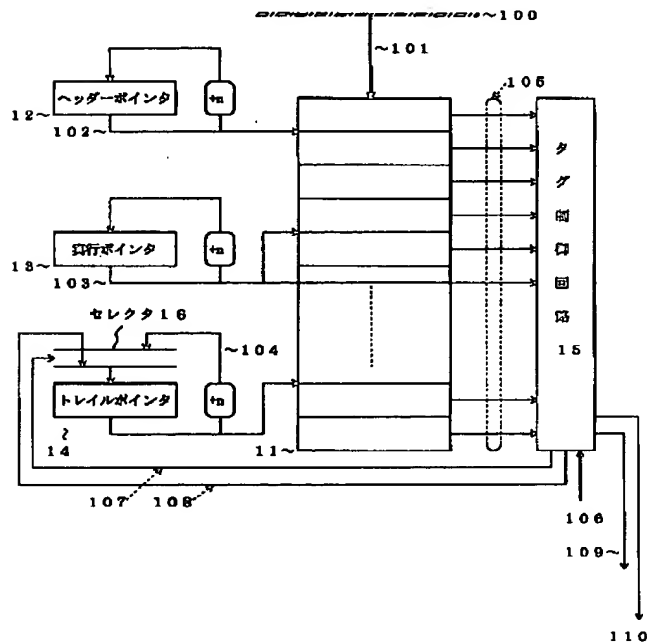
【図面の簡単な説明】

【図1】本発明のスーパー scaler 回路の一実施例を示すリオーダーバッファの外部構成図である。

【図2】本発明のスーパー scaler 回路の一実施例を示すソフトウェア命令デコード廻りのブロック図である。

【図3】本発明のスーパー scaler 回路のパイプラインキャンセル回路を示すブロック図である。

【図1】



【図5】

V	DestReg#	B	SrcTag#	Cc	Vr	WriteData	Status
---	----------	---	---------	----	----	-----------	--------

* 【図4】本発明のスーパー scaler 回路のファームウェア命令の展開イメージ図である。

【図5】本発明のスーパー scaler 回路のリオーダーバッファの登録内容である。

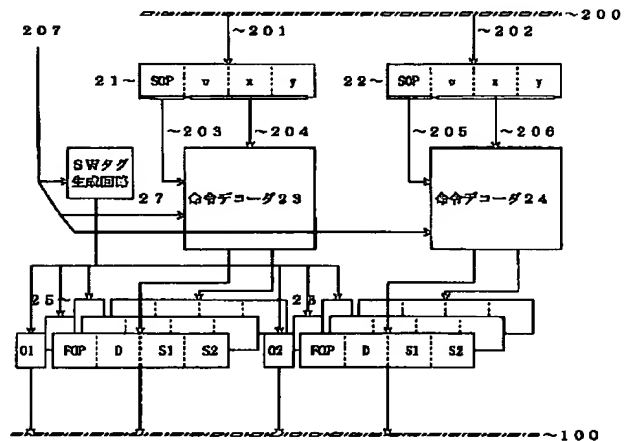
【図6】本発明のスーパー scaler 回路の図4のファームウェア命令列のリオーダーバッファの登録内容である。

【図7】従来例のリオーダーバッファの構成を示すブロック図である。

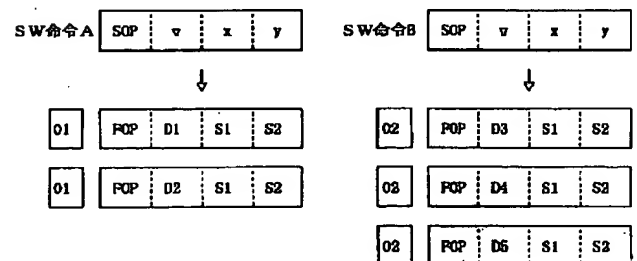
【符号の説明】

- 11 リオーダーバッファ
- 12 ヘッダーポインタ
- 13 実行ポインタ
- 14 トレイルポインタ
- 15 タグ制御回路
- 21, 22 ソフトウェア命令キュー
- 23, 24 命令デコーダ
- 25, 26 ファームウェア命令キューとSWタグ番号
- 27 SWタグ生成回路
- 31, 32 SWタグ番号
- 33, 34 タグチェック回路
- 35, 36 ステージバリッド
- 37, 38 論理積

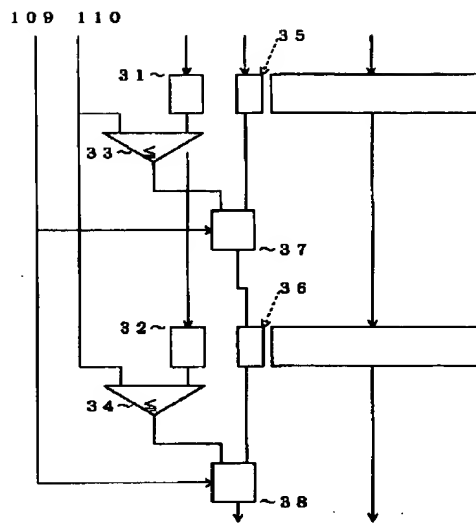
【図2】



【図4】

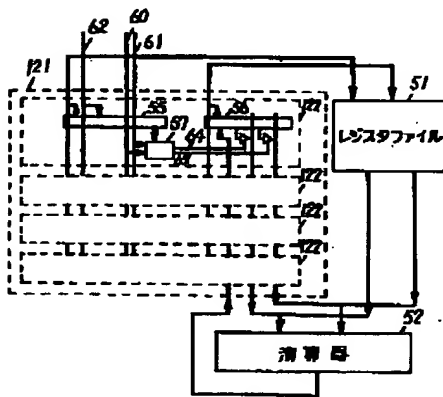


【図3】



【図7】

55 デスティネーションレジスタフィールド
 56 データフィールド
 57 検出器



【図6】

	V	DestReg#	B	Swtag#	Cc	Fv	WriteData	Status
20#	1	D1	0	01	-	1	01234567	Done
	1	D2	1	01		0		exec
	1	D3	1	02		0		exec
	1	D4	1	02		0		Ready
	1	D5	1	02		0		Wait